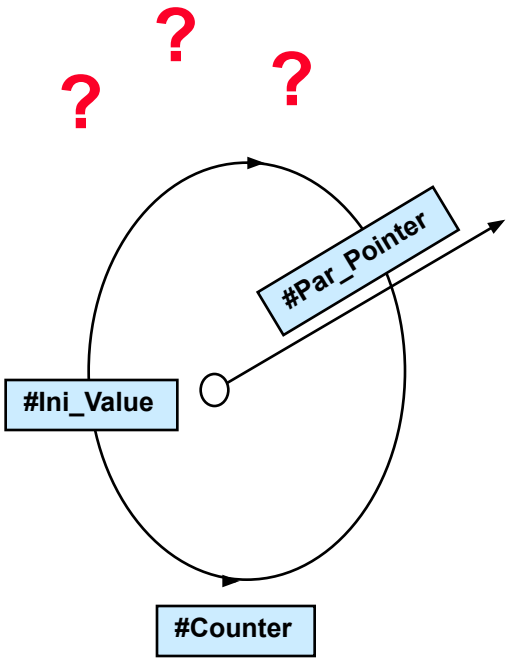
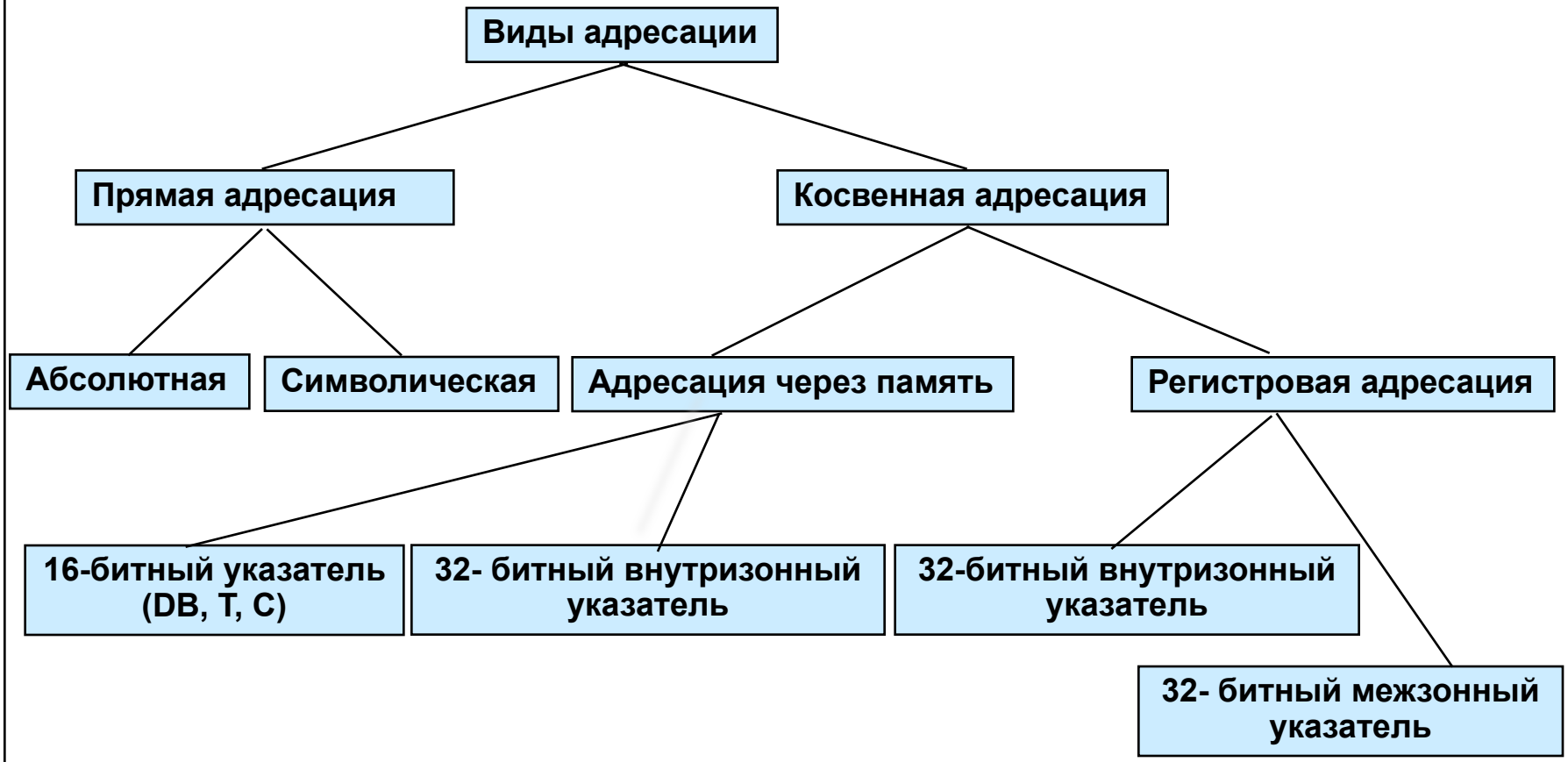
Косвенная адресация и инструкции с адресными регистрами



1
2
-
-
-
-
9
10

L W [AR1, P#200.0]

Виды адресации, доступные в STEP 7



Примеры:

```
A I 4.0       A "Mot_on" OPN DB[MW10]  A I[MD30]   A I[AR1,P#0.0]    A [AR1,P#0.0]
L IW10        L #Num     SP  T["runtime"]  L IW["Number"]   L ID[AR1,P#5.0]  T W[AR1,P#0.0]
```

Прямая адресация переменных

Адрес	Местоположение в памяти (например).	Ширина доступа	Значение
I	37.4	Байт, слово, двойное слово	Входы
Q	27.7	Байт, слово, двойное слово	Выходы
PIB	655	Байт, слово, двойное слово	Периферийные входы
PQB	653	Байт, слово, двойное слово	Периферийные выходы
M	55.0	Байт, слово, двойное слово	Меркеры
T	114	--	Таймеры
C	13	--	Счетчики
DBX	2001.6	Байт (DBB), слово (DBW), двойное слово(DBD)	Данные адресуются через DB регистр
DIX	406.1	Байт (DIB), слово(DIW), двойное слово(DID)	Данные адресуются через DI регистр
L	88.5	Байт (LB), слово (LW), двойное слово(LD)	Локальный стек

Адресные идентификаторы прямой адресации для DB

Открыть блок данных

```
OPN DB 19
OPN "Values"

OPN DI 20
```

Загрузка и перенос в блоках данных

```
L DBB 1  Загрузить байт данных 1
L DBW 2  Загрузить слово данных 2 (байты 2 и 3)
L 5   Загрузить число 5
T DBW 4  Перенести в слово 4
L 'A' Загрузить ASCII-символ A
L DIB28   Загрузить байт данных 28
==I   Сравнить

A DBX 0.0       Опросить бит 0 из байта 0
```

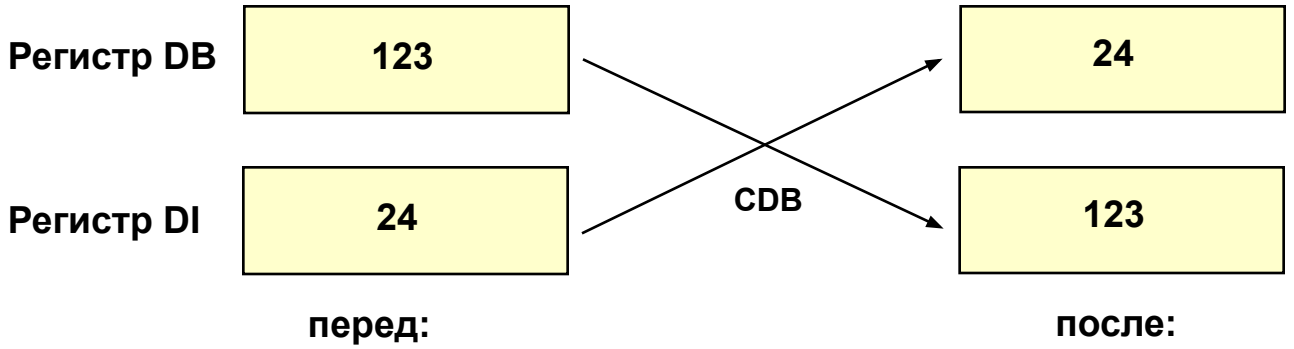
Комбинация инструкций (содержит OPN DB..)

```
L DB19.DBW4  Загрузить слово данных 4 из DB 19

L "Values".Number_1   Символический доступ к
                переменной Number_1. DB19,
                имеющей символьное имя
                "Values"
A DB10.DBX4.7       Опросить бит 7 из байта 4  DB 10
```

Оценка информации о DB в программе

Инструкции с регистрами DB:

- **CDB: Обмен содержимого DB - регистров**



- **Загрузить DB-регистр в ACCU1**
  - L DBNO (загрузить номер открытого DB в ACCU1)
  - L DINO (загрузить номер открытого DI в ACCU1)
- **Загрузить длину блока данных**
  - L DBLG (загрузить длину (в байтах) блока данных, открытого через DB, в ACCU1)
  - L DILG (загрузить длину (в байтах) блока данных, открытого через DI, в ACCU1)

Косвенная адресация через память

- **16-битный указатель в формате слова (адресация DB,T,C)**

```
L 11
T MW 60
L    C[MW60]          L    C11
LC T [MW60]           LC T11
OPN DB[MW 60]         OPN DB 11
UC FC[MW60]           UC FC11
CC FC[MW60]           CC FC11
UC FB[MW60]           UC FB11          CC FB[MW60]
CC FB11
```

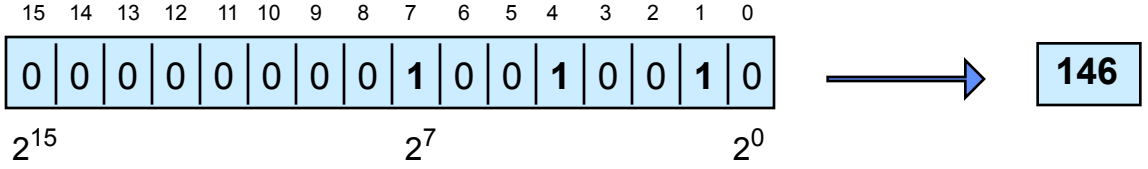
- **32- битный указатель в формате двойного слова (адресация I, Q, M, ...)**

```
L    P#24.0
T    MD 50

L    I    W    [MD50]      L    IW 24
```

Область — Ширина доступа — адрес

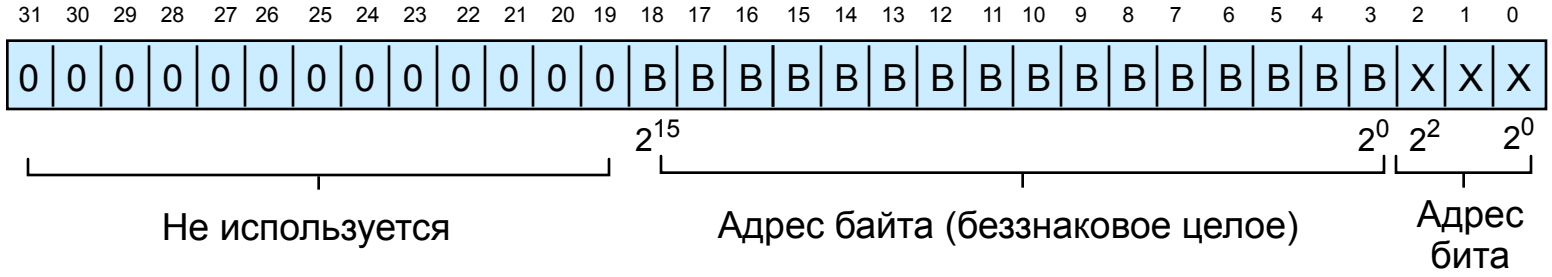
Структура указателя при косвенной адресации через память

- **Структура 16-битового указателя:**



Интерпретируется как беззнаковое целое число в интервале 0 ... 65 535

- **Структура 32-битового указателя (внутризонного):**



- **Загрузка константы типа 32-битового указателя (внутризонного):**

L P#25.3 (P = Pointer (указатель), Адрес байта= 25, Адрес бита= 3)

Специальные особенности косвенной адресации через память

Области памяти для сохранения 16- и 32-битовых указателей:

- Меркеры (адресуются абсолютно или символически,
  напр.:OPN DB[MW30], OPN DI["Motor_1"], и т.д.
  A I[MD30], T QD["Speed_1"], и т.д.)
- Локальный стек данных (адресуются абсолютно или символически,
  напр.: OPN DB[LW10], OPN DI[#DB_NO], и т.д.
  A I[LD10], T QD[#Par_Pointer], и т.д.)
- Глобальный (общий) блок данных (адресация можент быть только абсолютной, DB должен быть предварительно открыт,
  напр.: OPN DB[DBW0] (переписывается регистр DB !!!), OPN DI[DBW22],
  напр.: A I[DBD10], T QD[DBD22], и т.д.)
- Экземпляр блока данных (адресация можент быть только абсолютной, DI должен быть предварительно открыт,
  напр.: OPN DB[DIW20], OPN DI[DIW0] (переписывается регистр DI !!!),
  напр.: A I[DID10], T QD[DID22], и т.д.)

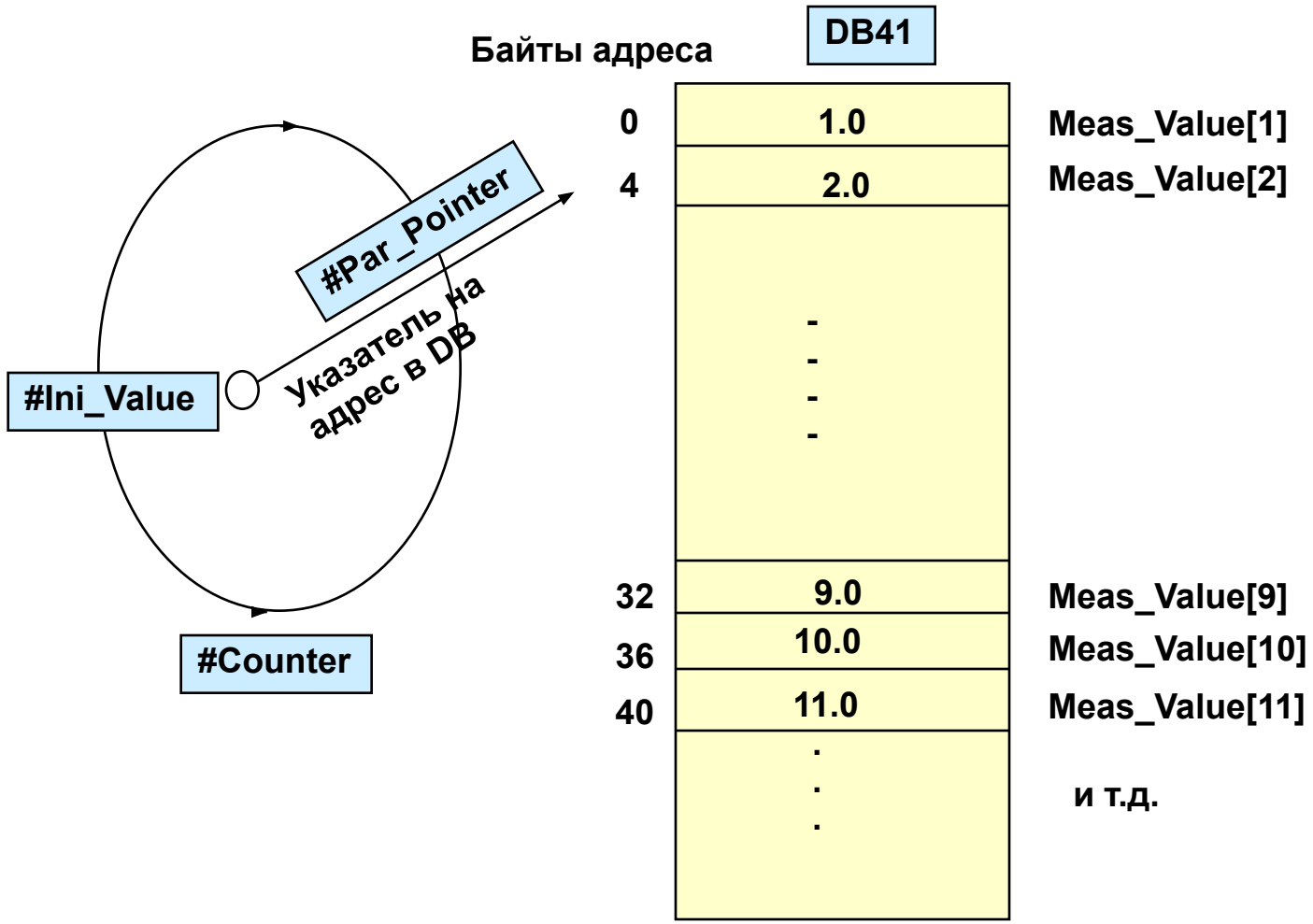
Характеристики в передачи указателей для FB и FC

- Указатели, используемые в параметрах, не могут использоваться непосредственно для косвенный адресации через память.
- Указатели для косвенной адресации, помещенные в память, перед вызовом должны быть скопированы во временные переменные.

Пример косвенной адресации

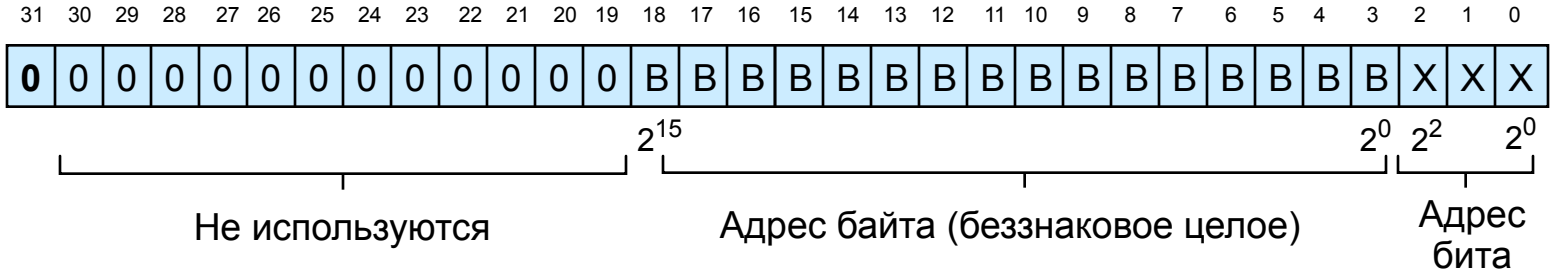
```
FC30:    Пример для косвенной адресации

Network 1:   Открыть DB с помощью косвенной адресации

     L    #dbnumber     // Скопировать номер DB в MW100
     T    MW   100 //
     OPN  DB[MW   100]  // Открыть DB


Network 2:   Цикл удаления

     L    P#18.0       // Сохранить конечный адрес(DBW18)как указатель
     T    MD   40      // в MD 40;
     L    10           // Установить счетчик цикла на 10
next:    T    MB   50      // и сохранить его в MB 50;
     L    0            // Загрузить инициализирующее значение
     T    DBW[MD   40]  // и перенести его в DB;
     L    MD   40      // Загрузить указатель,
     L    P#2.0         // уменьшить его на 2 байта
     -D                 // и перенести результат назад
     T    MD   40       // в MD 40;
     L    MB   50       // Загрузить счетчик цикла
     LOOP next          // Уменьшение счетчика и
                           // если,если он не равен 0, то переход;
```

Упражнение 4.1: Программирование цикла с косвенной адресацией

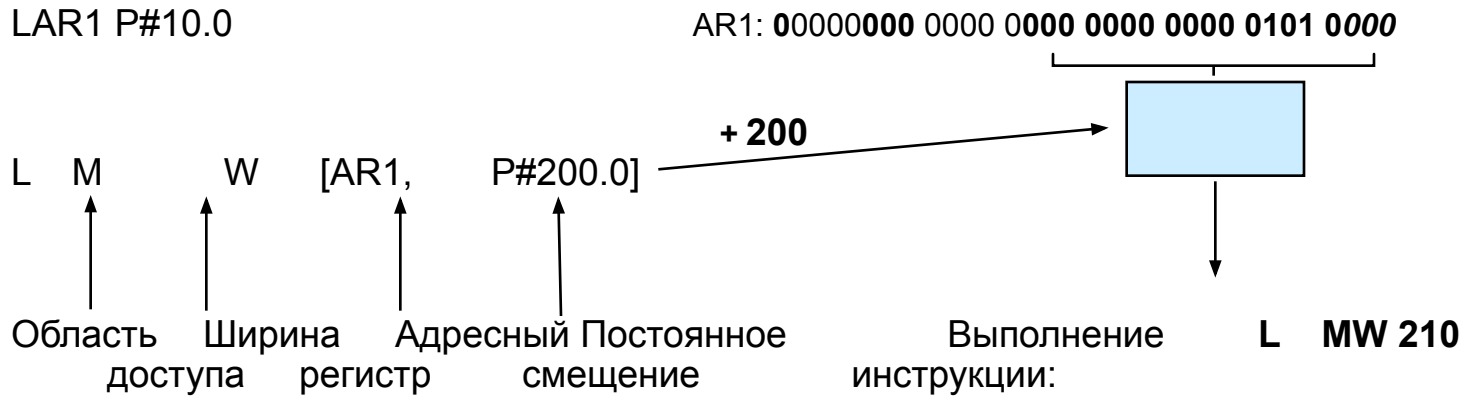


Внутризонная регистровая косвенная адресация

- **Внутризонный указатель в AR 1 или AR2:**

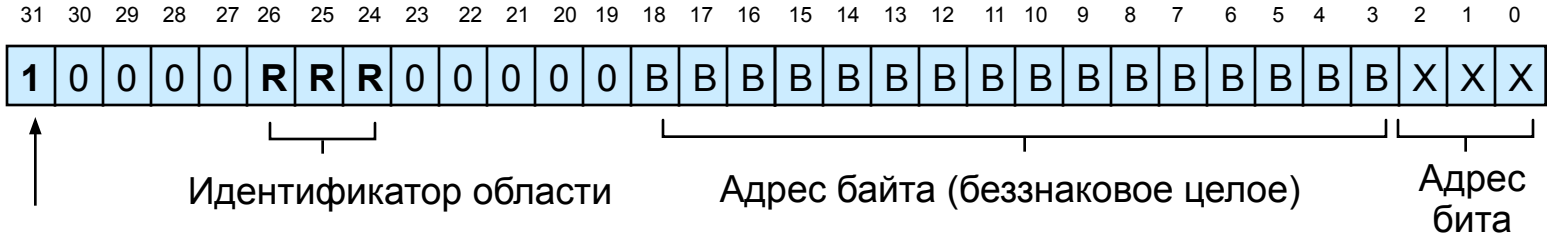


- **Синтаксис команды:**



Межзонная регистровая косвенная адресация

- **Межзонный указатель в AR 1 или AR2:**



Bit 31=0: внутризонная
Bit 31=1: межзонная

- **Идентификатор области:** 000 Периферия (P) 001 Входы (PII)
  010 Выходы (PIQ) 011 Память меркеров
  100 Блок данных, регистр DB 101 Блок данных, регистр DI
  110 Собственные локальные данные 111 Локальные данные вызывающего блока

- **Синтаксис команды:**

LAR1 P#I10.0

AR1: **1**0000**011** 0000 0**000 0000 0000 0101 0*000***

L W [AR1, **M** P#200.0] **+ 200**

Ширина доступа — Адресный регистр — Постоянное смещение

Выполнение инструкции: **L MW 210**

Инструкции для загрузки адресных регистров

Загрузка адресных регистров

- LARn (n =1 or 2): Загрузить содержимое ACCU1 в ARn
- LARn <Address> Загрузить содержимое <Address> в ARn
- LARn P#<Address> Загрузить адрес <Address> в ARn

<Address>:

- Регистры процессора: AR1, AR2 (напр., *LAR1 AR2* and *LAR2 AR1*)
- 32-битовые переменные: MDn, LDn, DBDn, DIDn (напр., *L DBD5*, и т.д.)
- символн. 32- битовые переменные : 32- битовые глобальные переменные (напр., *LAR1 "Index"*, и (глобальные и локальные) т.д.)
  и TEMP (временные) переменные OB, FB и FC
  (напр., *LAR1 #Address*, и т.д.)

P#<Address>

- Указатель с абсолютной битовой адресацией: En.m, An.m, Mn.m, Ln.m, DBXn.m, DIXn.m (напр., *LAR1 P#M5.3*, *LAR2 P#I3.6*, и т.д.)
- Указатель с локальной, символн. адресацией: OB: TEMP- переменные (напр.,: *LAR1 P##Par_Pointer*, и т.д.)
  FB: IN-, OUT-, INOUT-, STAT- и TEMP- переменные.
  FC: TEMP- переменные (*LAR1 P##Loop*, и т.д.)

Другие инструкции для адресных регистров

Перенос из адресного регистра

- TARn (n =1 or 2): Перенос содержимого из ARn в ACCU1
- TARn <Address> Перенос содержимого из ARn в <Address>

<Address>:

- Процессорные регистры: AR2 (напр., *TAR1 AR2*)
- 32 -битовые абс. переменные: MDn, LDn, DBDn, DIDn (напр., *TAR2 MD5*, и т.д.)
- сиволич. 32 -битовые переменные: 32- битовые глобальные переменные (напр., *TAR1*
  (глобальн. и локальные) *"Index"*, и т.д.) и TEMP- переменные OB, FB и FC (напр., TAR1 #Address, и т.д.)

Обмен адресных регистров

- TAR Обмен содержимого адресных регистров AR1 и AR2

Adding to Address Register

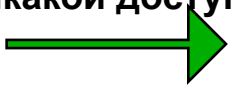
- +ARn Прибавить ACCU1-L к ARn
- +ARn P#x.m Прибавить указатель без указания области P#x.m к ARn

Специальные особенности адресных регистров

Внутреннее использование AR1 STL/LAD/FBD-редактором

- При доступе к параметрам в FC, используются **регистры AR1 и DB**, если параметры имеют сложный тип данных (ARRAY, STRUCT, DATE_AND_TIME).
- При доступе к INOUT-параметрам FB, используются **AR1 и DB регистры**, если INOUT- параметр имеет сложный тип данных (ARRAY, STRUCT, DATE_AND_TIME)

Никакой доступ к локальным параметрам не возможен



между командой загрузки в адресный регистр и командой косвенного доступа через регистр к желаемой переменной

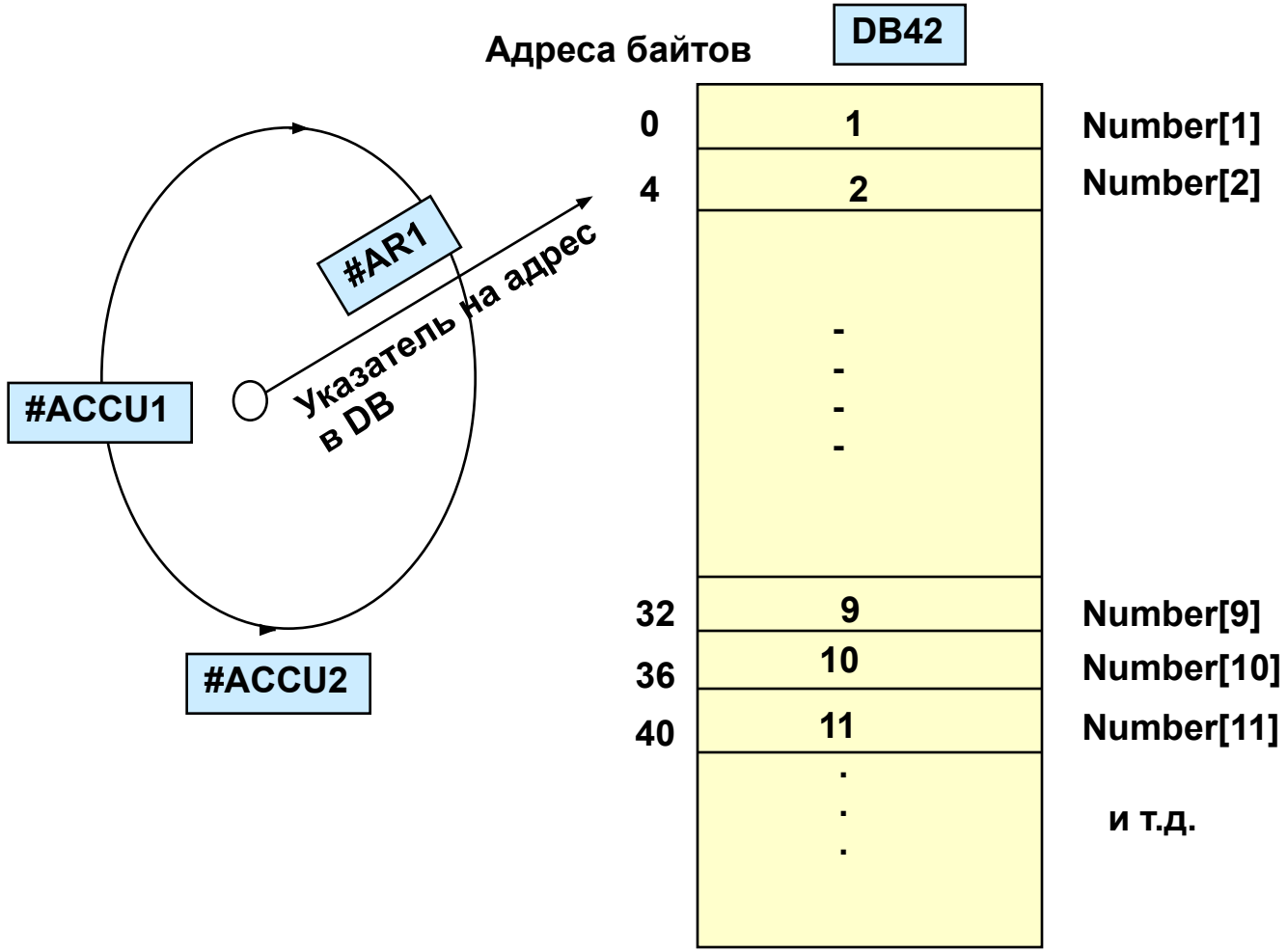
Внутреннее использование AR2 STL/LAD/FBD-редактором

- **Регистр AR2** и **регистр DI** используется как база адреса для адресации всех параметров и STAT-переменных в **FB**.

Если **AR2** или **DI** - изменяются пользователем внутри FB, никакой доступ к собственным параметрам или STAT-переменным не может иметь место без восстановления обоих регистров.

- Никаких ограничений в отношении регистра AR2 и регистра DI в пределах FC нет.

Упражнение 4.2: Программирование цикла с регистровой косвенной адресацией



Типы указателей в STEP 7

16-битовый указатель для косвенной адресации через память

- Для косвенного доступа через память к таймерам, счетчикам, для открытия блоков данных и для вызова FC без параметров и FB без параметров и STAT-переменны

32-битовый указатель для косвенной и регистровой адресации через память

- 32-битовый внутризонный указатель для косвенного доступа через память и регистры в области PI, PQ, I, Q, M, DB, DI и L (локальный стек данных)
- 32 -битовый межзонный указатель для косвенного доступа через регистры в области PI, PQ, I, Q, M, DB, DI, L и V (локальный стек данных вызывающего блока)

48-битовый указатель (тип данных: POINTER)

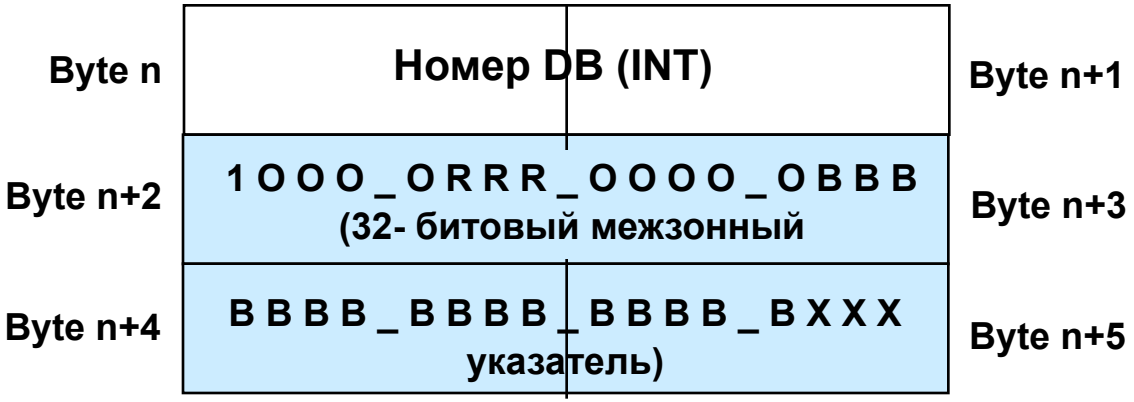
- Тип данных для передачи параметров в блоки (FB и FC)
- В дополнение к 32-битовому межзонному указателю содержит номер DB

80-битовый указатель (тип данных: ANY)

- Тип данных для передачи параметров в блоки (FB и FC)
- В дополнение к 32-битовому межзонному указателю содержит номер DB, тип данных и коэффициент повторения

Структура и назначение типа данных POINTER

Структура типа данных POINTER



Назначение параметров типа POINTER

- **Вид указателя**
  P#DBn .DBX x.y где: n= номер DB, x= номер байта, y= номер бита
  P#DIn .DIX x.y (напр.: P#DB5.DBX3.4, P#DI2.DIX10.0, и т.д.)
  P#Zx.y где: Z= область, напр.: P, I, Q, M и L
  (напр.: P#I5.3, P#M10.0, и т.д.)
- **Объявление адреса:**
  MD30 (в этом случае, номер DB , идентификатор
  #Motor_on области и битовый адрес автоматически
  "Motor_1".speed вводится в POINTER)

Структура типа данных ANY

- **Указатель ANY для типов данных**

Byte n	16#10	Тип данных
Byte n+2	Коэффициент повторения	
Byte n+4	Номер DB	
Byte n+6	1 O O O _ O R R R _ O O O O _ O B B B	
Byte n+8	B B B B _ B B B B _ B B B B _ B X X X	

Тип данных	Идентификатор
VOID	00
BOOL	01
BYTE	02
CHAR	03
WORD	04
INT	05
DWORD	06
DINT	07
REAL	08
DATE	09
TOD	0A
TIME	0B
S5TIME	0C
DT	0E
STRING	13

- **Указатель ANY для параметрических типов**

Byte n	16#10	Параметрический тип
Byte n+2	16#0001	
Byte n+4	16#0000	
Byte n+6	16#0000	
Byte n+8	Номер таймера, счетчика или блока	

Параметр. тип	Идентификатор
BLOCK_FB	17
BLOCK_FC	18
BLOCK_DB	19
BLOCK_SDB	1A
COUNTER	1C
TIMER	1D

Назначение параметров с типом данных ANY

Вид указателя:

- **P#[Data block.]Битовый адрес Числовой тип**

  P#DB10.DBX12.0 REAL 20 Указатель на область в DB10, начинающуюся с 12-го байта, содержащую 20 ячеек с типом данных REAL (ARRAY[1..20] OF REAL)

  P#I 10.0 BOOL 8 Указатель на область из 8 бит в IB10

Объявление адреса:

- **абсолютное:**

  DB5.DBD10 Тип данных: DWORD, коэффиц. повтор.(КП): 1 номер DB: 5, указатель: P#DB5.DBX10.0

  IW32 Тип: WORD, КП: 1, №DB: 0, указатель: P#I 32.0

  T35 Тип : TIMER, Номер.: 35

- **символическое:**

  #Motor_1.speed
  "Pump".Start для элементарных типов данных компилятор устанавливает корректный тип данных, коэффициент повторения 1 и указатель

Примечание

При символическом назначении (ARRAY, STRUCT, STRING, UDT) в указателе ANY компилятором установливается идентификатор типа данных 02 (BYTE) и длина области в байтах.

Косвенное назначение параметра типа ANY

Назначение фактического значения типа ANY временной переменной

- **объявление временной переменной типа ANY в вызываемом блоке**

  например: `temp  aux_pointer ANY`

- **заполнение временной переменной ANY информацией о указателе**

  например:

```
LAR1 P##aux_pointer // Загрузить адрес на aux_pointer
L    B#16#10        // Загрузить идентификатор 10
TLB [AR1,P#0.0] // и перенести его со смещением 0
L...
...
```

- **Назначение параметрам блока значения типа ANY (целевая область) с помощью вспомогательной переменной с указателем**

  например:

```
CALL  FC    111
   Targetfield:=#aux_pointer
```

Преимущество

- **Динамическое переназначение параметрам указателя ANY во время выполнения**

Использование переданного указателя ANY

Address	Declaration	Name	Type	Initial Value	Comment	
0.0	in	Par_Pointer	ANY			
	ut					
0.0	temp	Data_type	BYTE			
2.0	temp	WF	WORD			
4.0	temp	DB_Nr	WORD			
6.0	temp	Area_Pointer	DWORD			

```
Network 1: Выделение типа данных, коэффициента повторения, номера DB и
           указателя
     L    P##Par_Pointer // Загрузка адреса of #Par_Pointer в ACCU1
     LAR1                // и загрузка его в AR1;
     L    B [AR1,P#1.0]  // Выделение типа данных из указателя
     T    #Data_type     // и загрузка во временную переменную;
     L    W [AR1,P#2.0]  // Выделение коэффициента повторения
     T    WF             // и загрузка во временную переменную;
     L    W [AR1,P#4.0]   // Выделение номера DB
     T    #DB_Nr          // и загрузка во временную переменную;
     L    D [AR1,P#6.0]   // Выделение указателя
     T    #Area_Pointer   // и загрузка во временную переменную;
```

Упражнение 4.3: Функция вычисления суммы и среднего значения